

# TP Individual: Middlewares y Coordinación de Procesos

Documentación Técnica



75.74 Sistemas Distribuidos I

Apellido, Nombre	Padrón	e-mail
Pérez, Martín Nicolás.	97378	manperez@fi.uba.ar

## Tabla de Contenidos

<b>75.74 Sistemas Distribuidos I</b>	<b>1</b>
<b>Introducción</b>	<b>3</b>
<b>Alcance</b>	<b>4</b>
<b>Visión general de la Arquitectura de Software</b>	<b>5</b>
<b>Objetivos de Arquitectura y Restricciones</b>	<b>6</b>
Objetivos	6
Restricciones	6
<b>Vista Lógica</b>	<b>7</b>
Diagramas de Clases	7
Diagramas de Estado	10
<b>Vista de Procesos</b>	<b>11</b>
Diagramas de Secuencia	12
Diagramas de Actividades	13
<b>Vista de Desarrollo</b>	<b>19</b>
Diagramas de Paquetes	19
<b>Vista Física</b>	<b>22</b>
Diagrama de Robustez	22
Diagrama de Despliegue	24
<b>Escenarios</b>	<b>25</b>
Diagrama de Casos de Uso	25
Descripción	26
<b>Simulación</b>	<b>27</b>

# Introducción

Este documento pretende exponer la solución para el desarrollo del Trabajo práctico individual de la materia Sistemas Distribuidos I de la Facultad de Ingeniería de Buenos Aires.

Se busca resolver un problema de procesamiento de datos, obtenidos de fuentes externas, mediante un sistema distribuido el cual procesará la información y devolverá los resultados solicitados.

Aquí se documentará las decisiones de diseño y arquitectura para la implementación de este sistema, las distintas vistas arquitectónicas, escenarios posibles, y las diferentes hipótesis y restricciones que se deben tener en cuenta para la posible solución a implementar.

# Alcance

El sistema será capaz de responder las consultas de diferentes clientes que cargarán los datos a procesar, generando los resultados del procesamiento y enviárselos a los clientes para que estos puedan visualizarlos.

El sistema procesará información de videos de YouTube y generará los siguientes ítems para el output del cliente.

- Los **pares únicos** de (id, title, category) de los videos etiquetados como 'funny' con más de 10M de likes.
- La **descarga del thumbnail** de aquellos videos que fueron trending por al menos 3 semanas en todos los países al mismo tiempo.
- El **día con más vistas** totales sobre los vídeos que recibieron más de 10M de likes durante ese día.

La consulta del cliente será realizada a través de una terminal desde su computador para luego esperar sus resultados que serán mostrados por pantalla en la misma terminal. El sistema garantizará que los clientes puedan acceder al contenido descargable en su máquina local una vez realizada la consulta.

# Visión general de la Arquitectura de Software

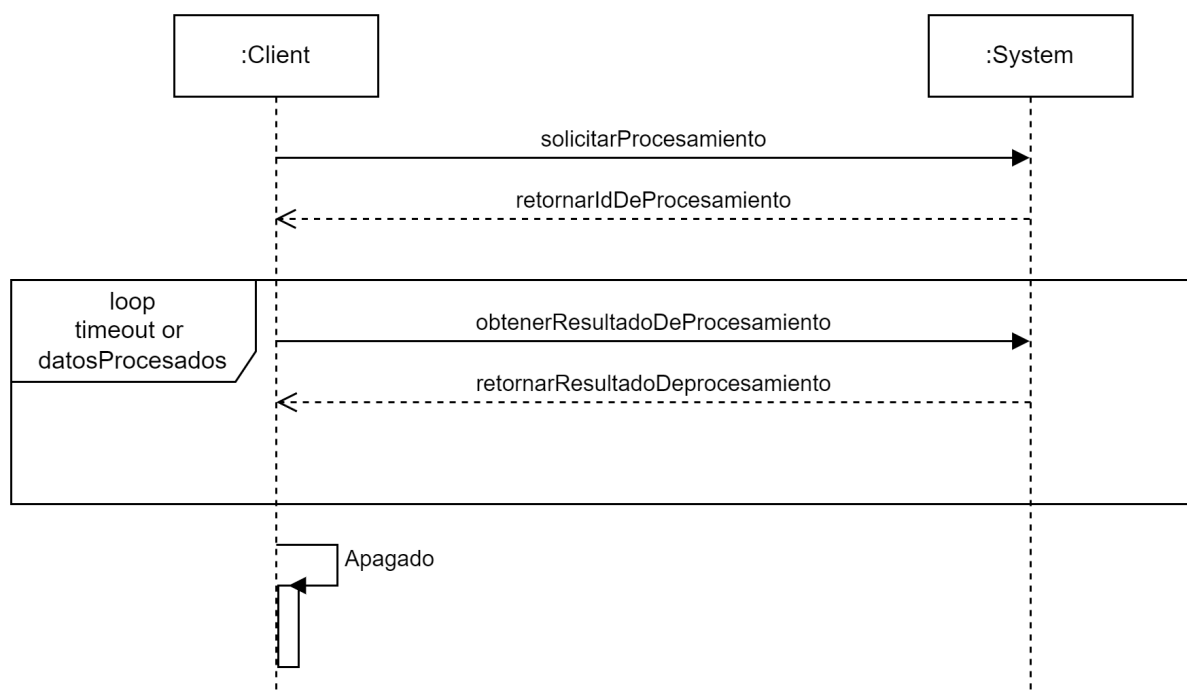
Los usuarios finales realizan la consulta a través de una aplicación **cliente**, la cual estará instalada en la máquina local del usuario, conectándose al **sistema** e ingresando la información que se requiere procesar.

El sistema estará compuesto por diferentes aplicaciones distribuidas comunicadas a través mensajes capaces de realizar operaciones sobre los datos de entrada generando resultados parciales. El flujo de datos se comparte de aplicación en aplicación, a través de mensajes, hasta poder generar los resultados esperados los cuales serán enviados finalmente al cliente solicitante.

Los clientes realizarán una llamada de procedimiento remoto (RPC) solicitando el procesamiento de datos utilizando el patrón de comunicación Request Reply de operación asíncrona. De esta manera, el cliente solicitará el proceso de la información en la primera comunicación e irá solicitando la obtención de datos periódicamente hasta que estos estén procesados completamente.

La comunicación será orquestada por un **Middleware**, que ofrece las interfaces correspondientes, tanto al cliente como a los distintos elementos del sistema, para enviar y recibir mensajes comunicando los distintos estados y eventos posibles. Para la implementación del envío de mensajes se utiliza el Framework **RabbitMQ**.

El siguiente diagrama muestra simplificadaamente la secuencia de comunicación entre el cliente y el sistema en la solicitud del procesamiento de datos.



# Objetivos de Arquitectura y Restricciones

## Objetivos

El diseño de esta arquitectura busca exponer una solución a un problema de procesamiento de datos mediante un **sistema distribuido**. Dicho sistema se compone de un conjunto de **aplicaciones** distribuidas en distintas unidades de cómputo o procesadores (Modelo Multi Computing) las cuales se comunican mediante **mensajes** enviando y recibiendo la información necesaria para el correcto y completo procesamiento de datos.

Esta arquitectura tiene como objetivos.

- Paralelizar la carga de trabajo en unidades de cómputo
- Orquestar el procesamiento de la información mediante mensajes
- Soportar el incremento de los elementos de cómputo para escalar los volúmenes de información a procesar
- Abstractar la complejidad de la comunicación basada en grupos a través de un Middleware
- Procesar la información de múltiples clientes

## Restricciones

La arquitectura propuesta está atada a las siguientes restricciones.

- Los datos ingresados deben estar previamente recolectados por un cliente
- La recolección de datos debe estar separada por País
- Se debe definir un **protocolo de comunicación**, utilizando mensajes, para indicar el inicio y fin del ingreso de la información
- El cliente al ingresar los datos debe especificar sobre qué País fueron recolectado dichos datos

# DAG

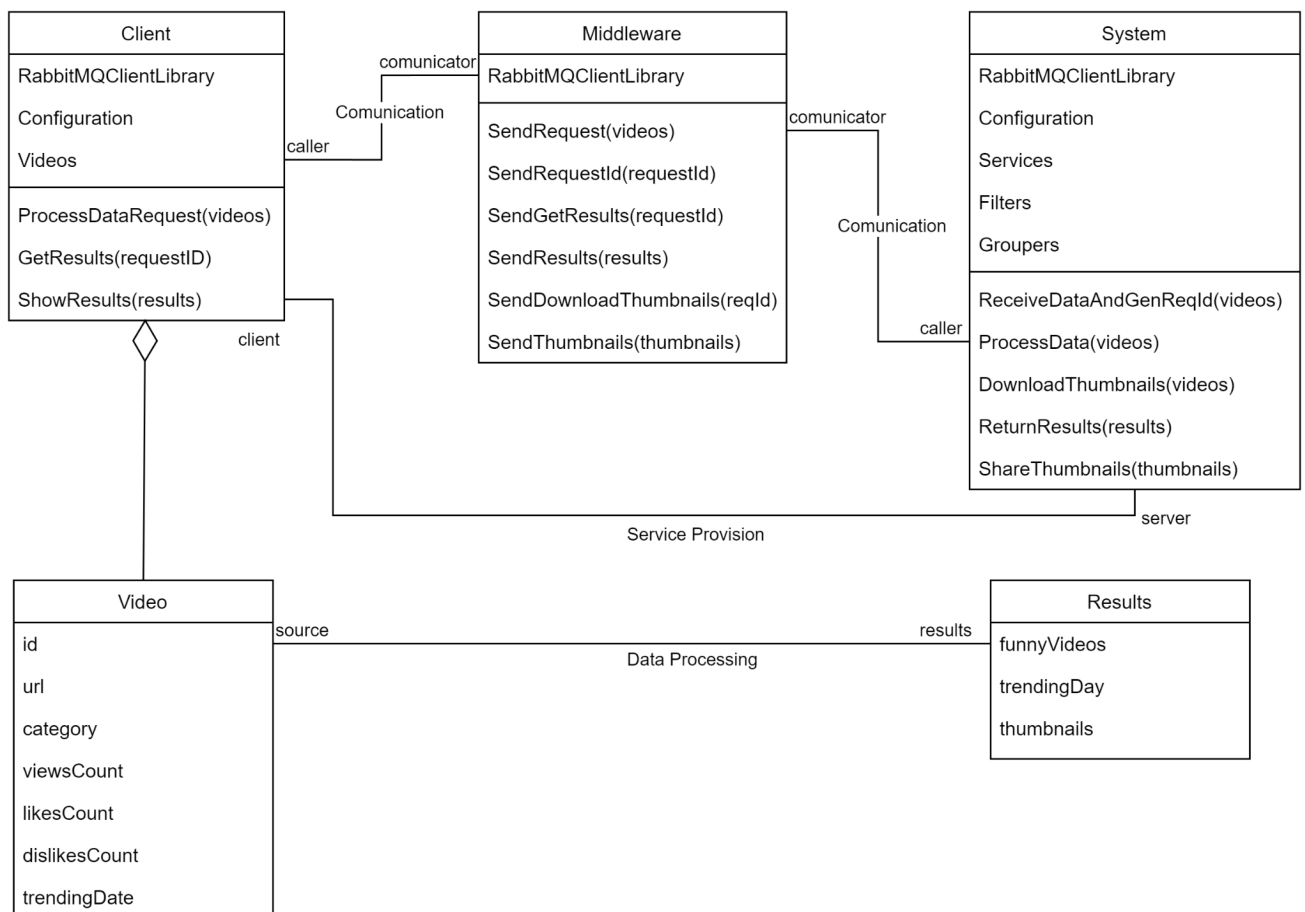
## Vista Lógica

En esta sección se pretende documentar la vista lógica del sistema, representando la funcionalidad que el sistema proporcionará a los usuarios finales, sus funciones y servicios que ofrece.

## Diagramas de Clases

### Diagrama de Clases 1 - Entidades

Entidades básicas que intervienen en el procesamiento de datos.



## Diagrama de Clases 2 - Sistema

Aquí se detalla la composición del sistema, con sus distintos servicios, filtros y agrupadores, así como también se expone la relación de comunicación entre todas las entidades del sistema.

El sistema está compuesto por una serie de servicios (**Services**), un conjunto de filtros (**Filters**) y un agrupador (**Grouper**).

### **Services**

- IngestionService
- ResultService
- DownloadThumbnailService
- ReportingService

### **Filters**

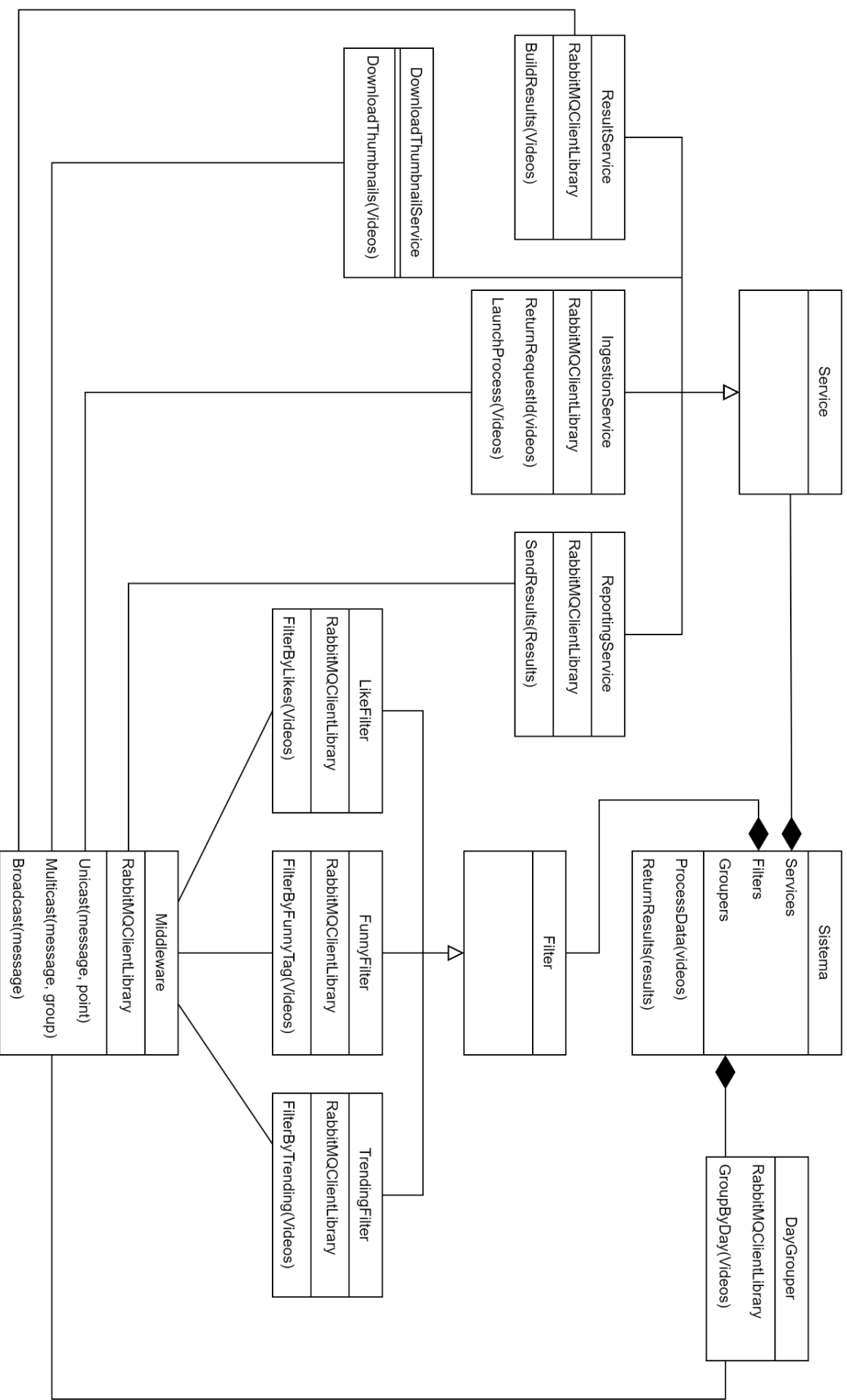
- LikesFilter
- FunnyFilter
- TrendingFilter

### **Grouper**

- DayGrouper

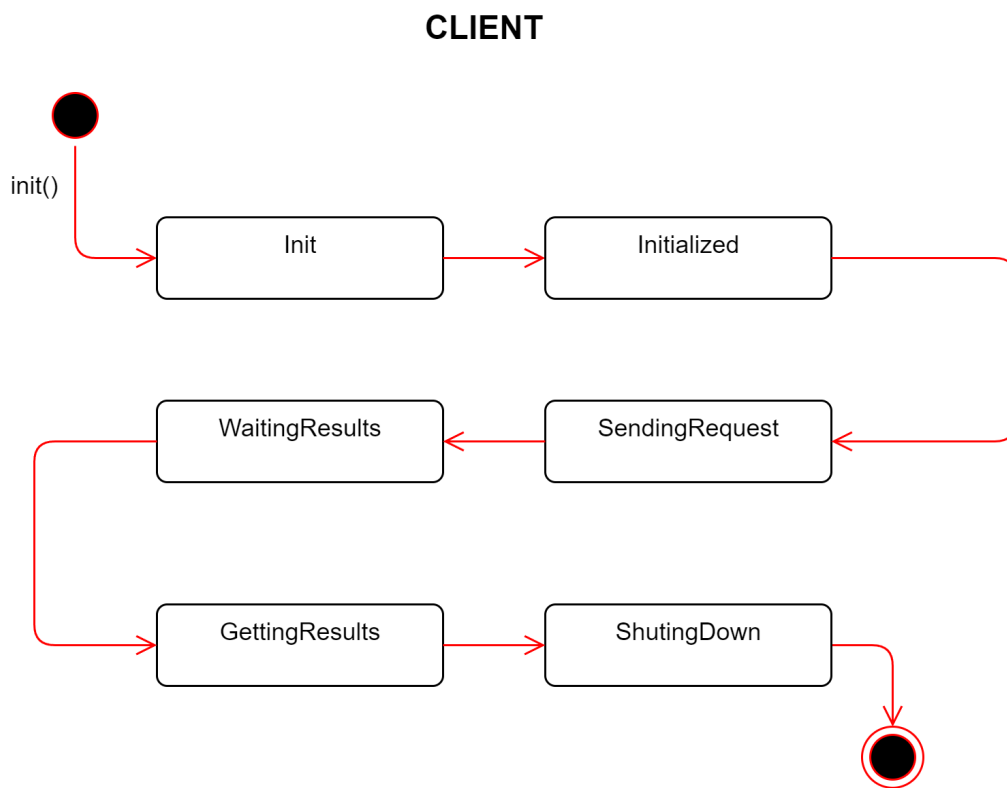
La comunicación entre los distintos componentes del sistema se coordinará a través del **Middleware** utilizando grupos de mensajes.





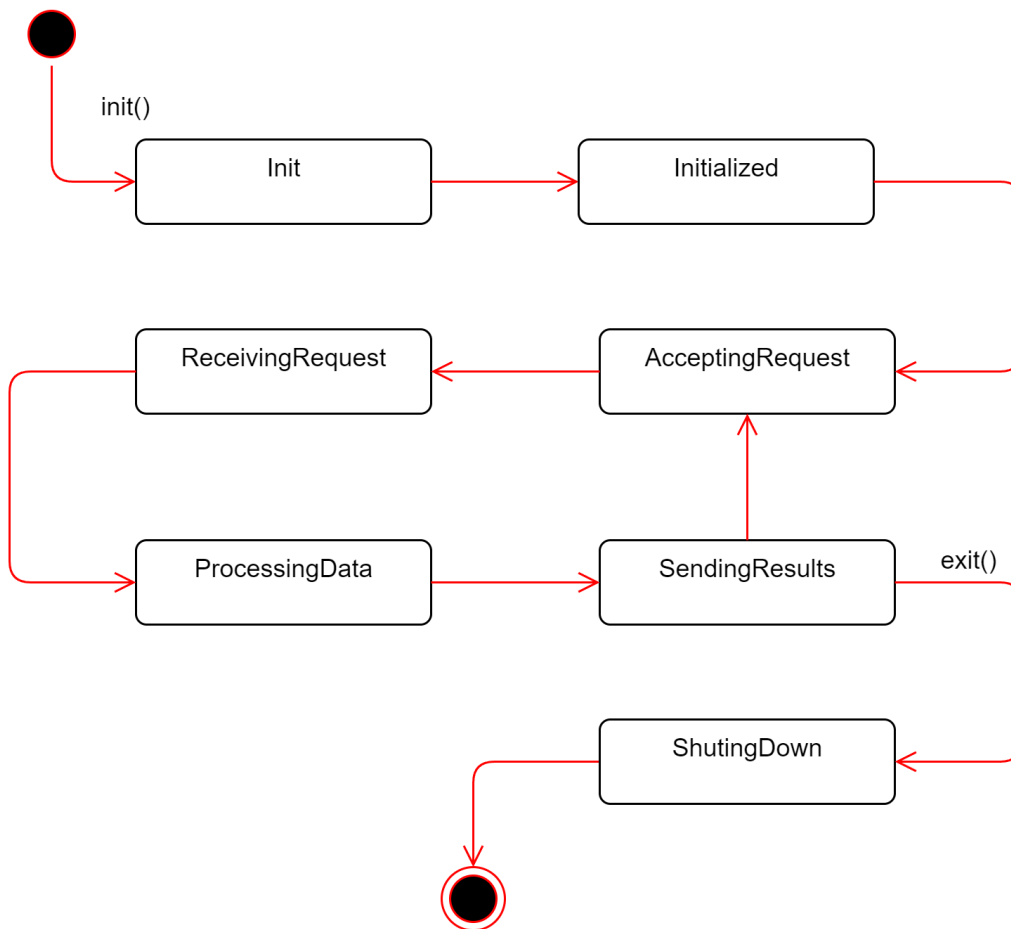
# Diagramas de Estado

## Diagrama de Estados (Cliente)



## Diagrama de Estados (Sistema)

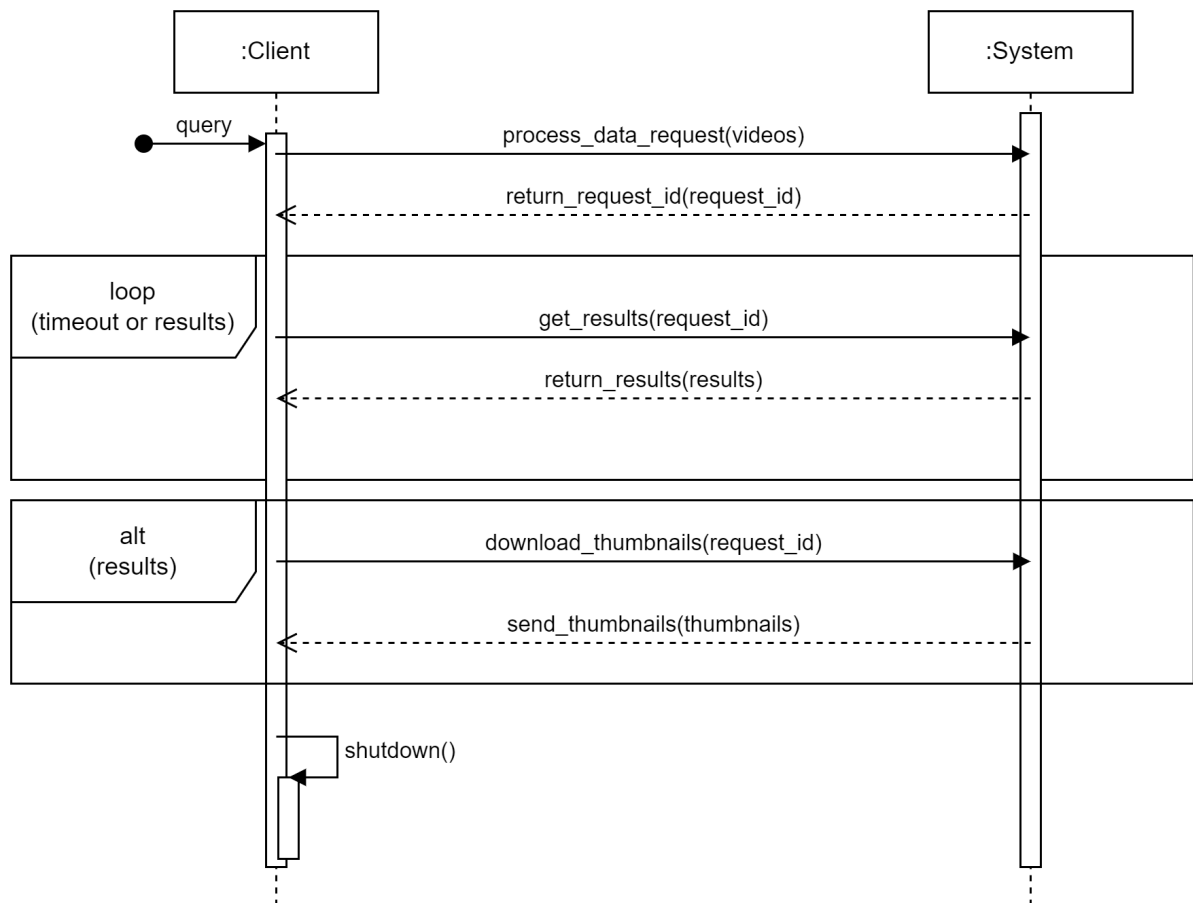
### SYSTEM



## Vista de Procesos

En esta sección se pretende documentar la vista de procesos del sistema, explicando su comportamiento detallando los distintos procesos, cómo se comunican y sincronizan la información resultante.

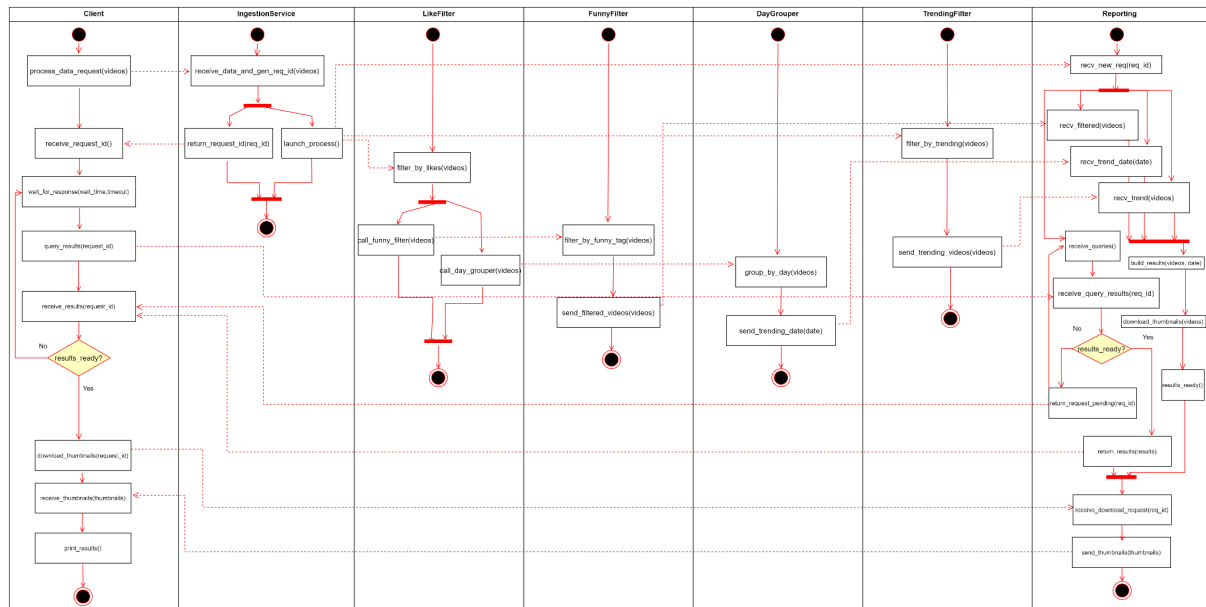
## Diagramas de Secuencia



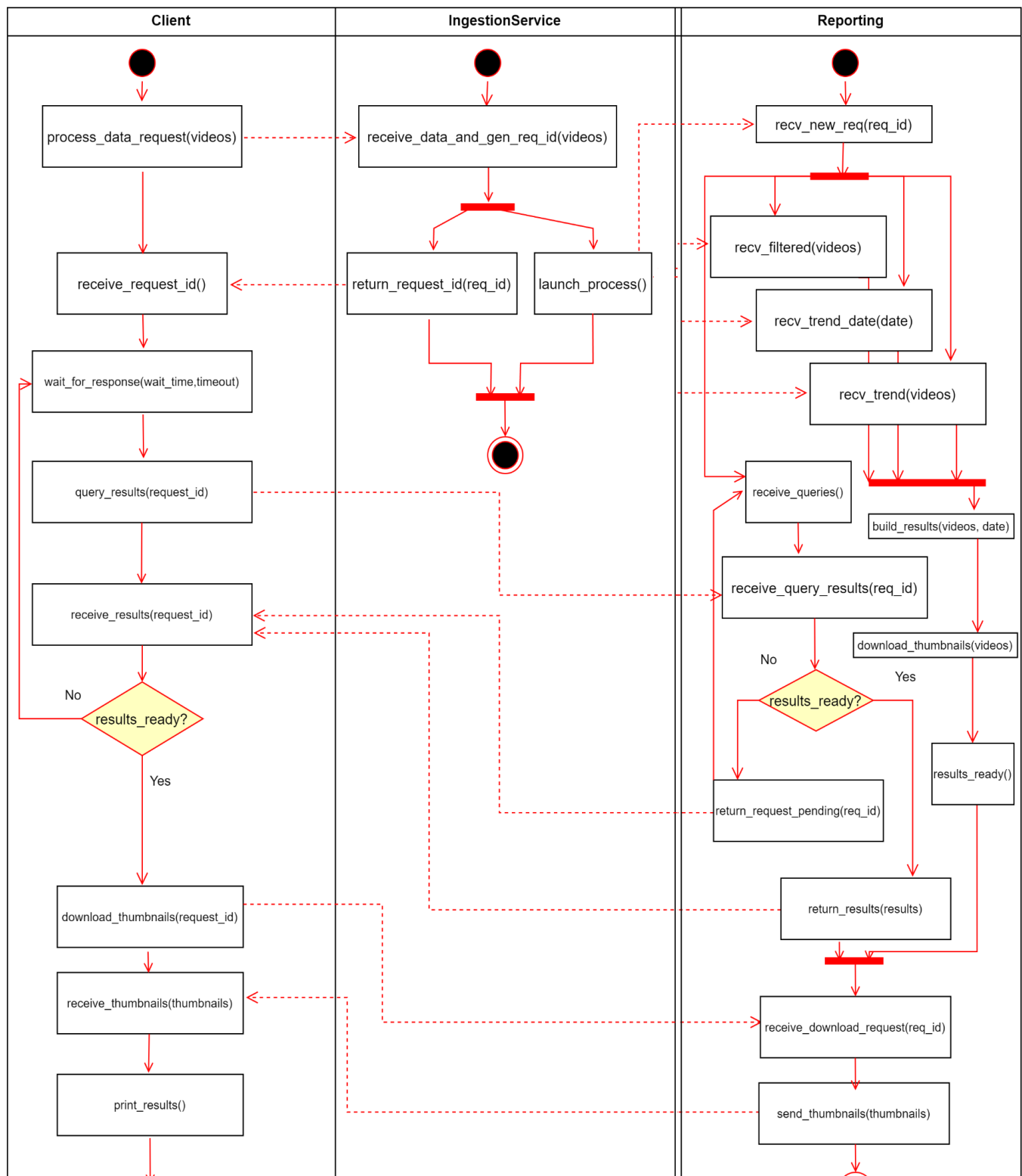
# Diagramas de Actividades

El siguiente diagrama de actividad se separó en diferentes partes para dar mejor visibilidad.

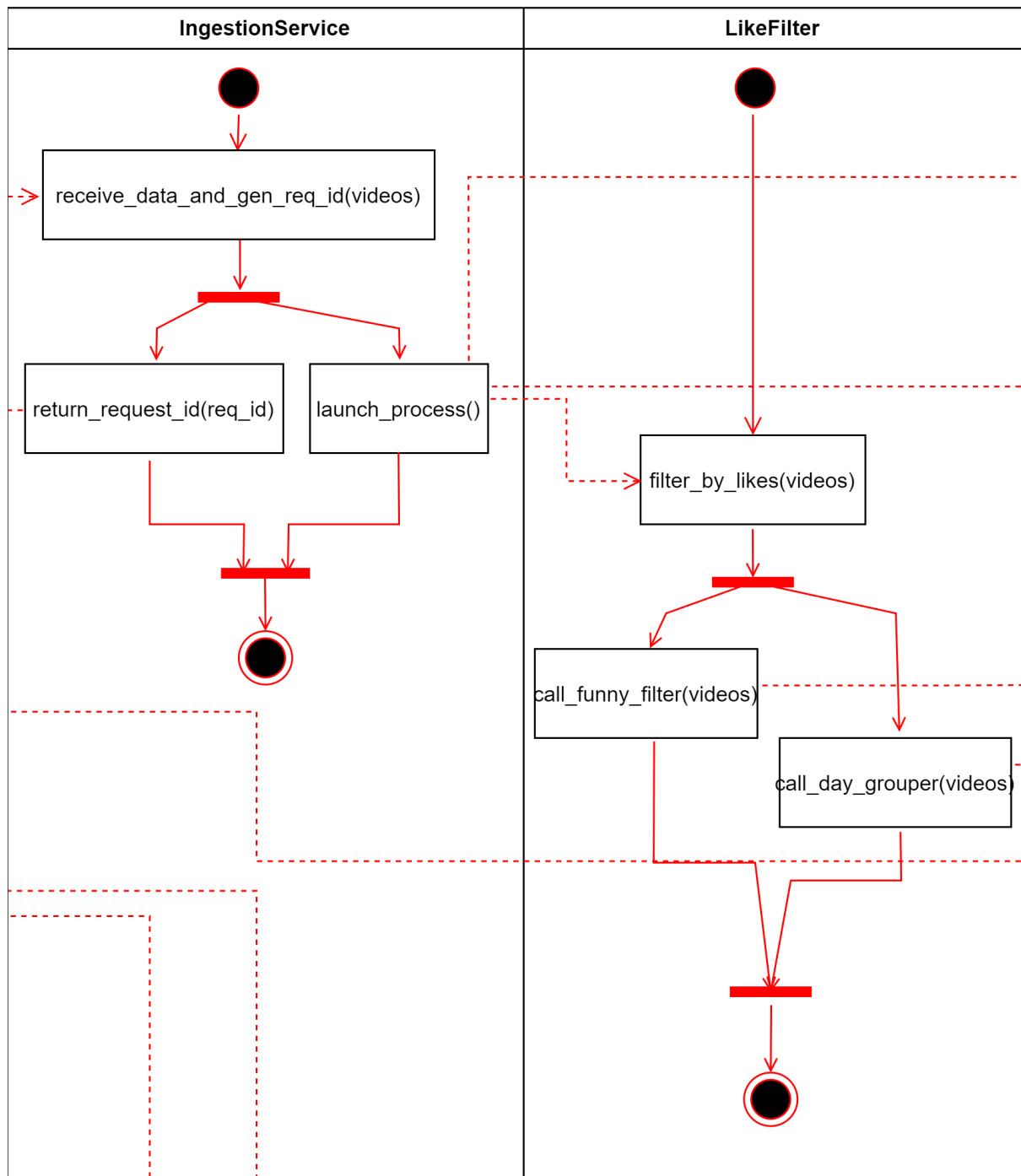
## Diagrama de Actividades - Completo



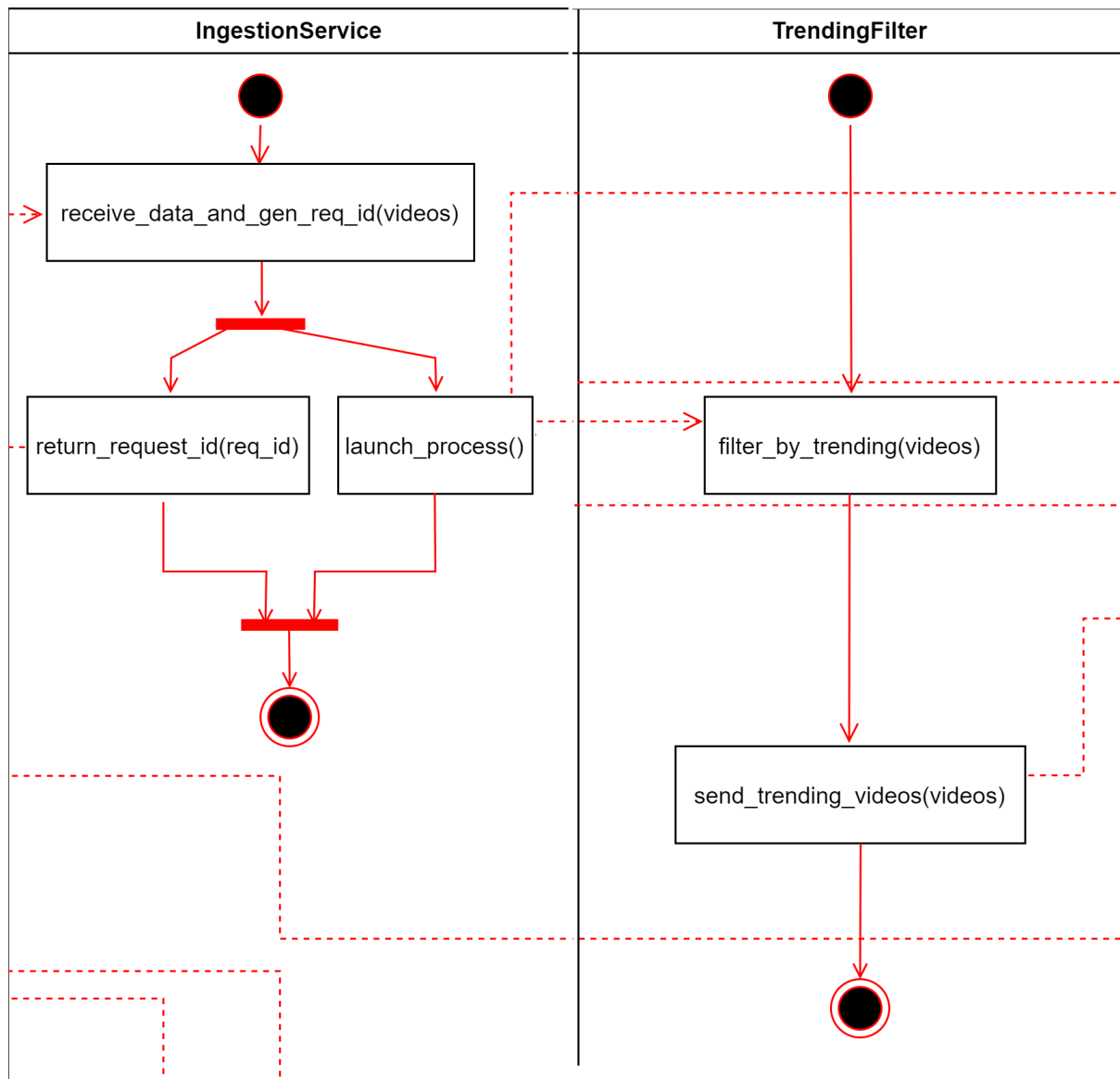
## Diagrama de Actividades - Ingesta de información y Reporte de resultados



## Diagrama de Actividades - Ingesta de información y Ejecución (Parte 1)

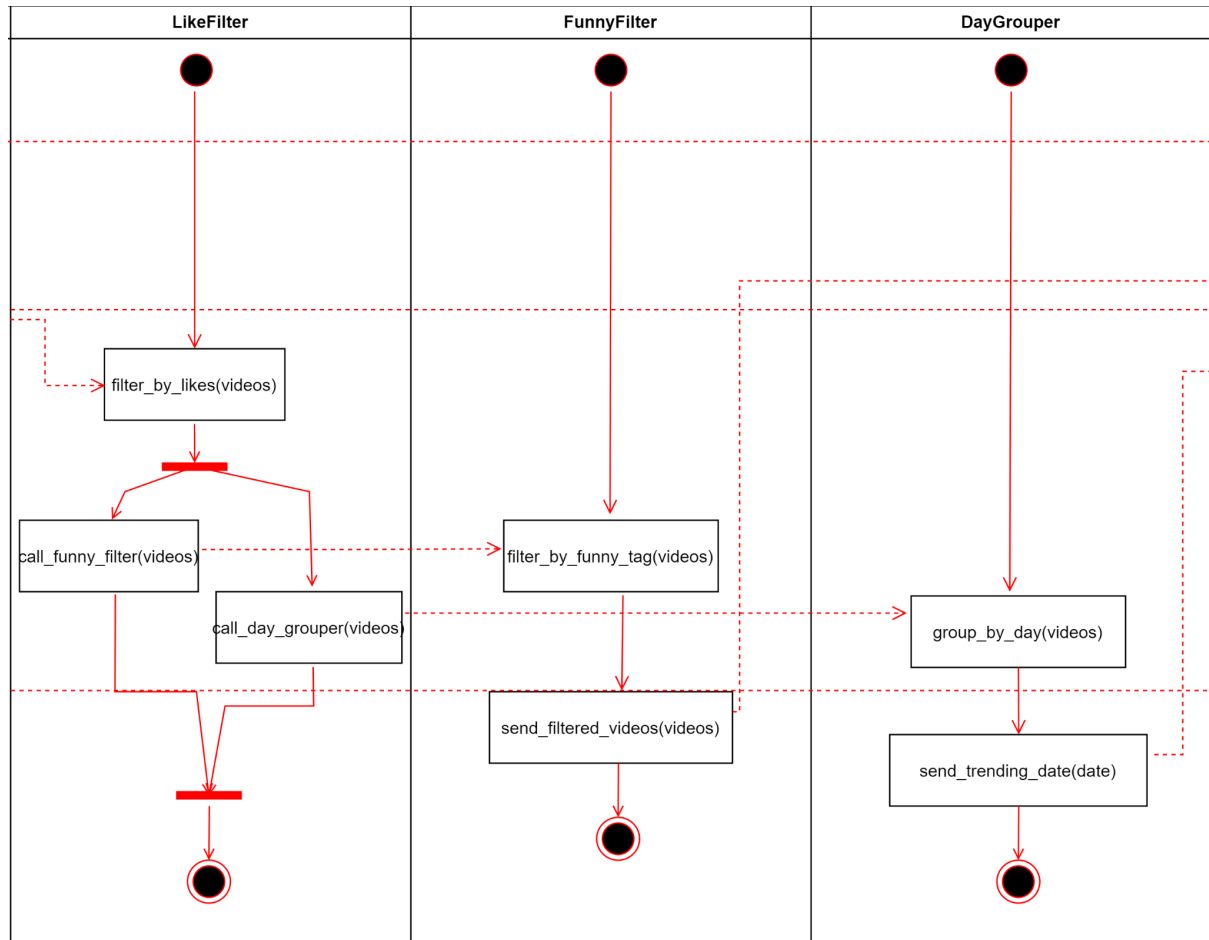


## Diagrama de Actividades - Ingesta de información y Ejecución (Parte 2)

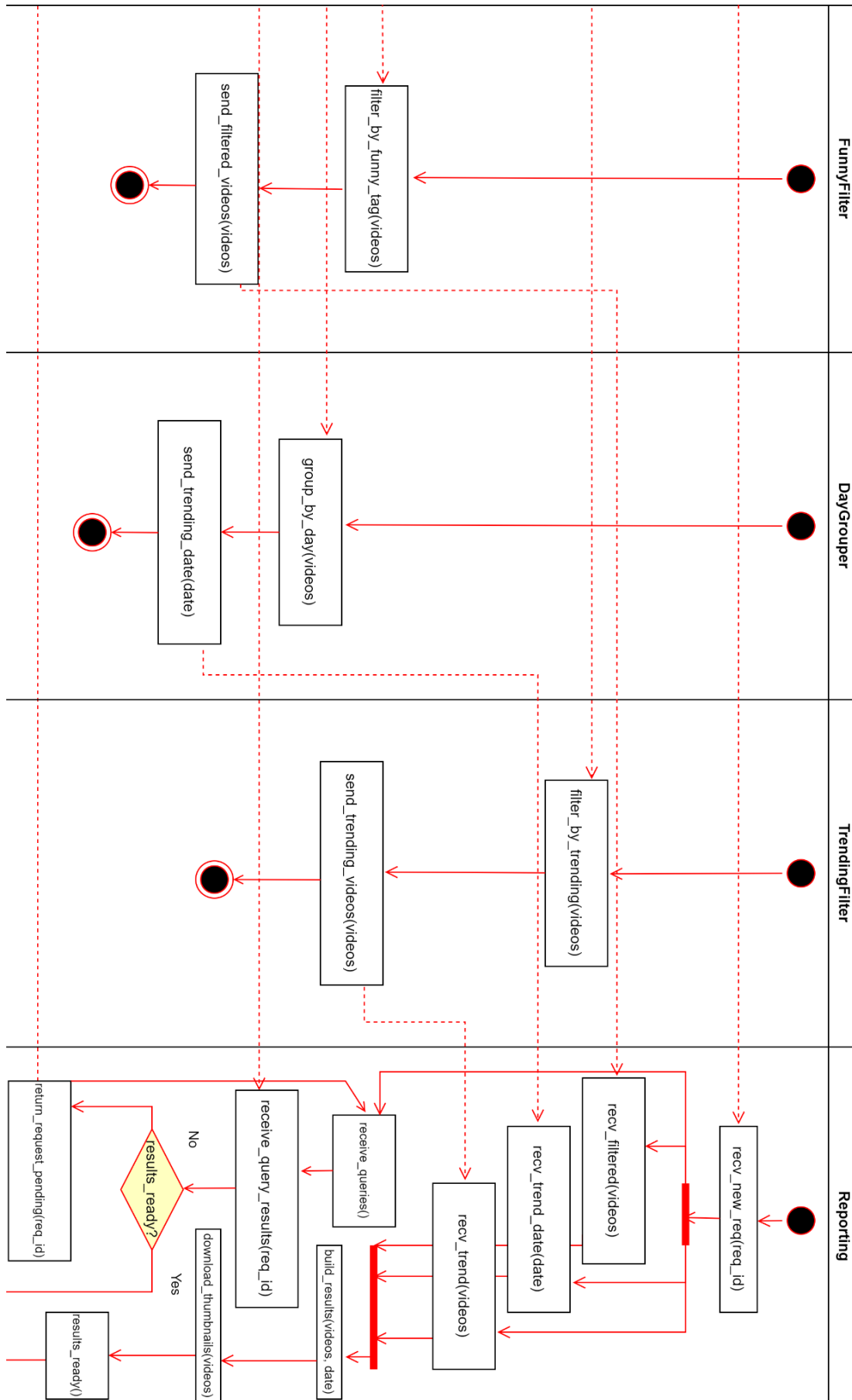




### Diagrama de Actividades - Ingesta de información y Ejecución (Parte 3)



### Diagrama de Actividades - Procesamiento de la información



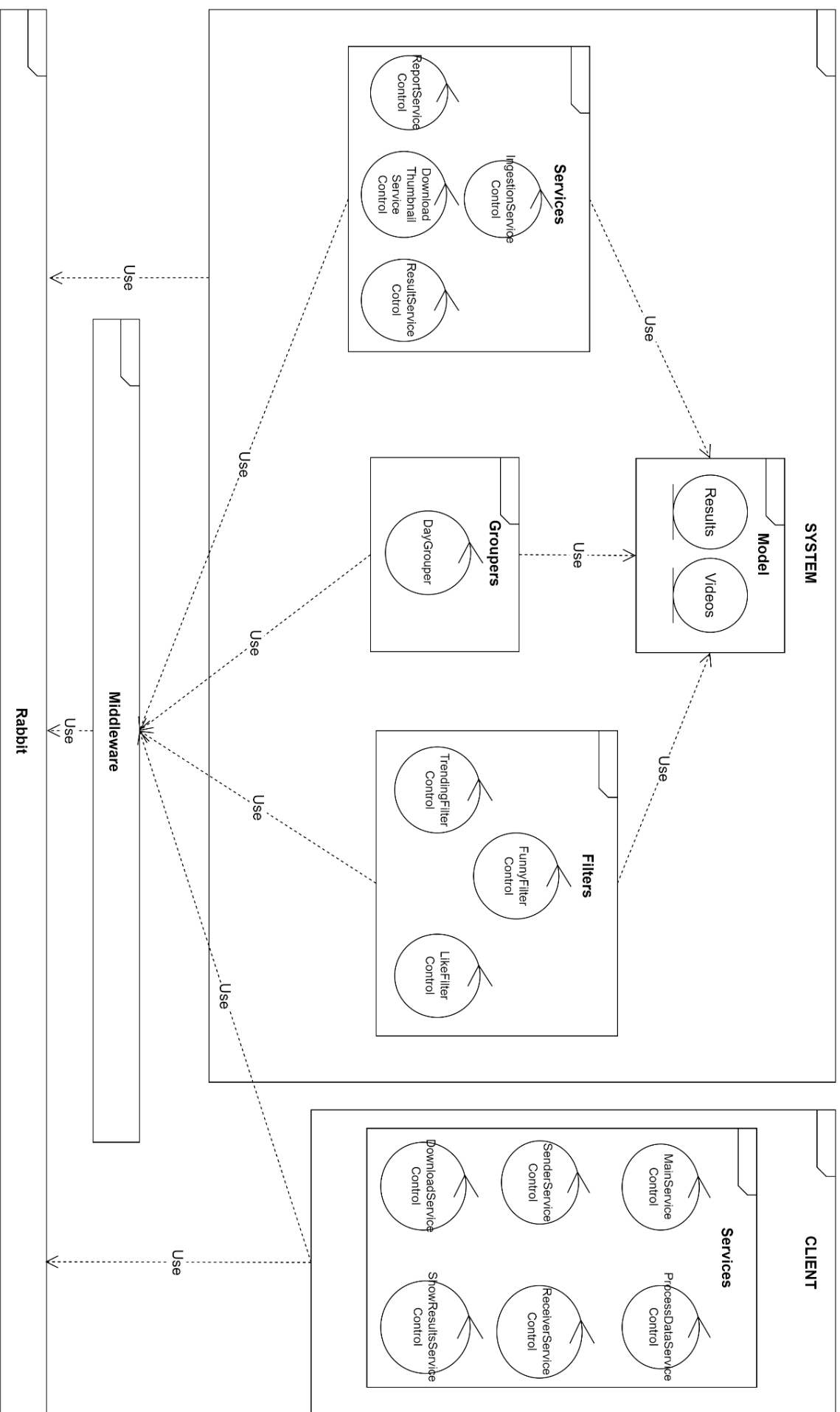
# Vista de Desarrollo

En esta sección se pretende documentar la vista de desarrollo describiendo los artefactos de software involucrados, la división del sistema, sus capas y las dependencias entre ellas. Así como también la definición de las aplicaciones distribuidas que encapsulan las funcionalidades y características del sistema.

## Diagramas de Paquetes

### Diagrama de Paquetes 1 - Funcionalidad

El siguiente diagrama representa las distintas capas y dependencias del Cliente, Sistema y Middleware. Muestra cuales son las distintas capas de *funcionalidad* tanto en el Cliente como en el Sistema.

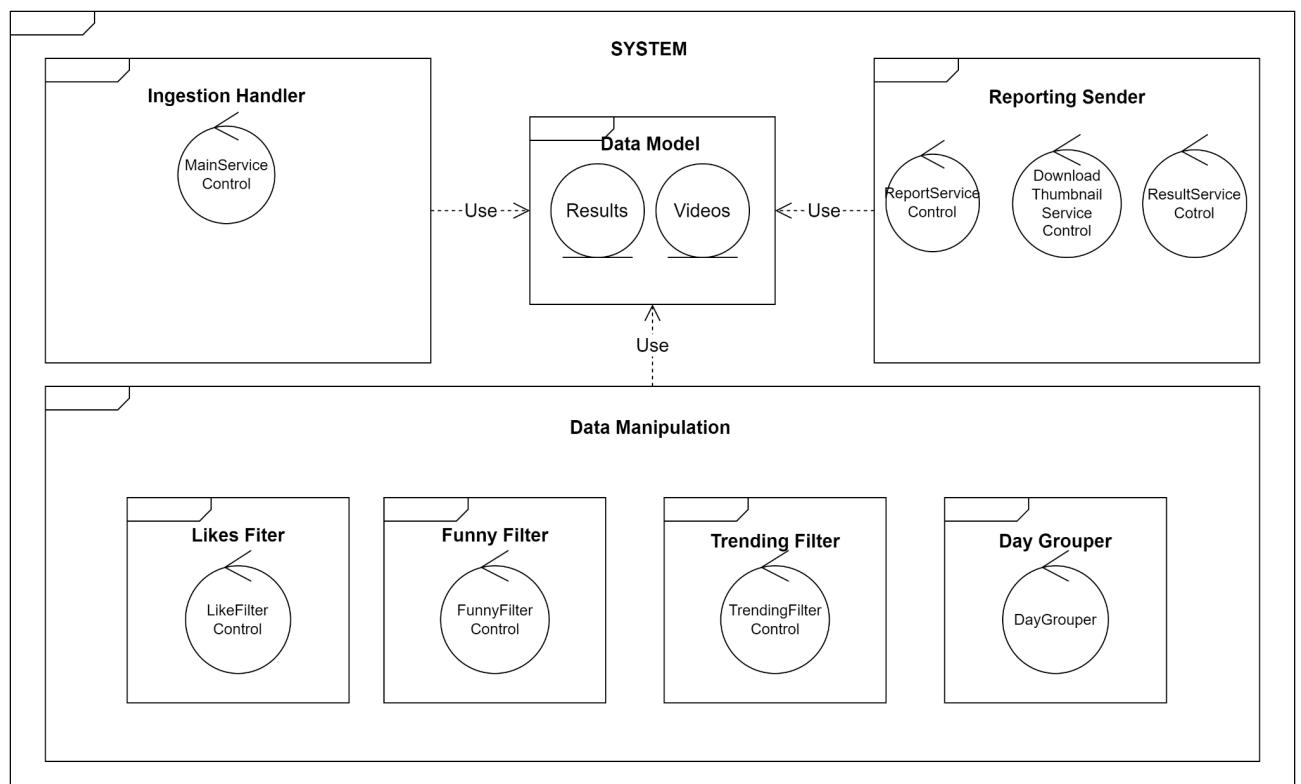


## Diagrama de Paquetes 2 - Módulos

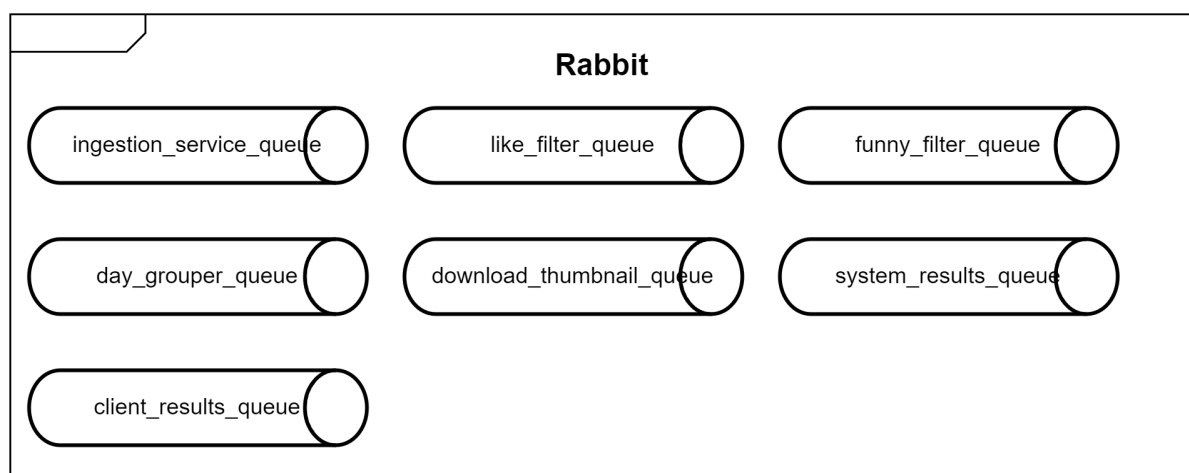
El Sistema posee 4 *Módulos* principales

1. Data Model
2. Ingestion Hanlder
3. Reporting Sender
4. Data Manipulation

Cada una de las aplicaciones distribuidas empaquetará uno de los módulos o submódulos del Sistema.



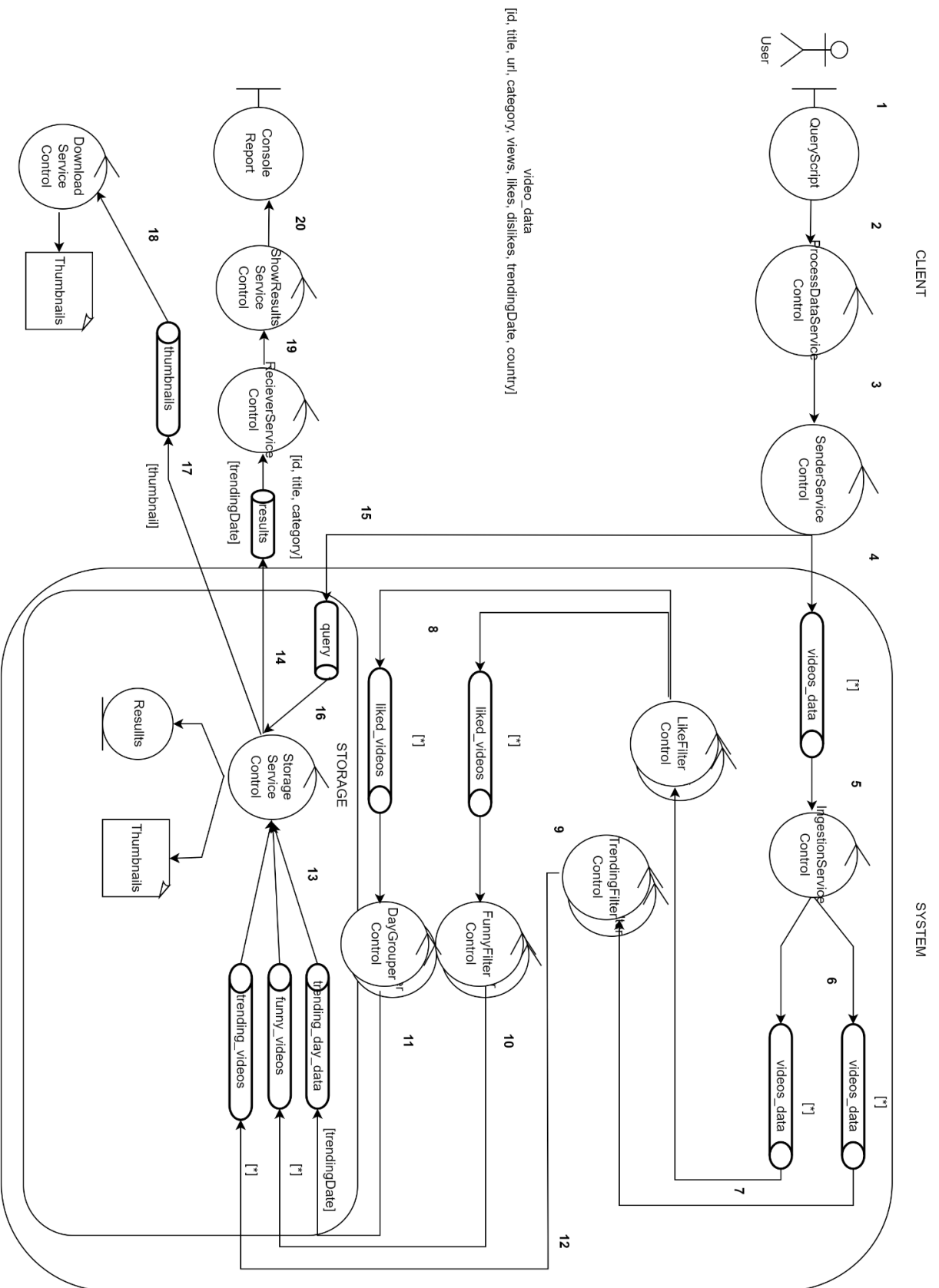
## Diagrama de Paquetes 3 - Definición de colas de Rabbit



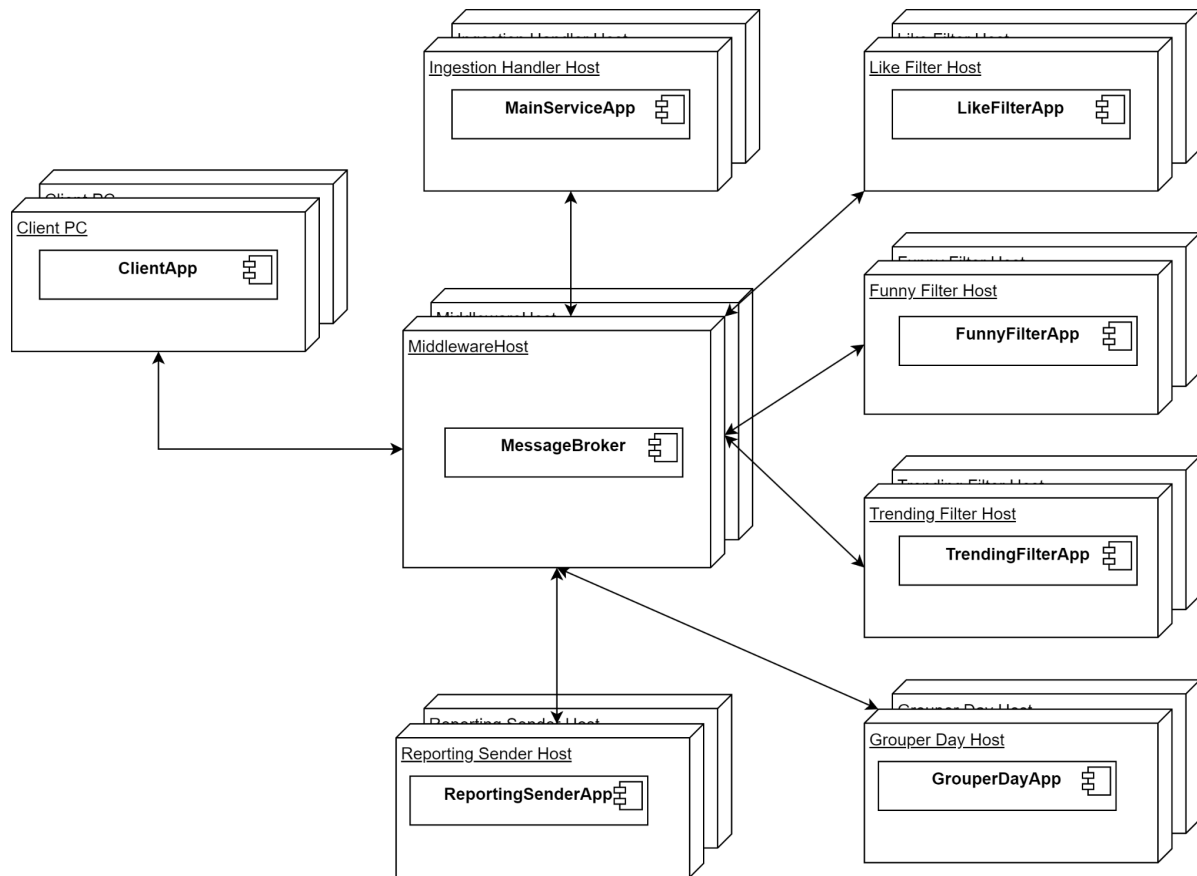
## Vista Física

En esta sección se pretende documentar la vista física del sistema exponiendo su arquitectura, conexión y comunicación de los distintos servicios. Así como también la distribución de las distintas aplicaciones o softwares involucrados en distintos hardwares/hosts.

## Diagrama de Robustez



## Diagrama de Despliegue

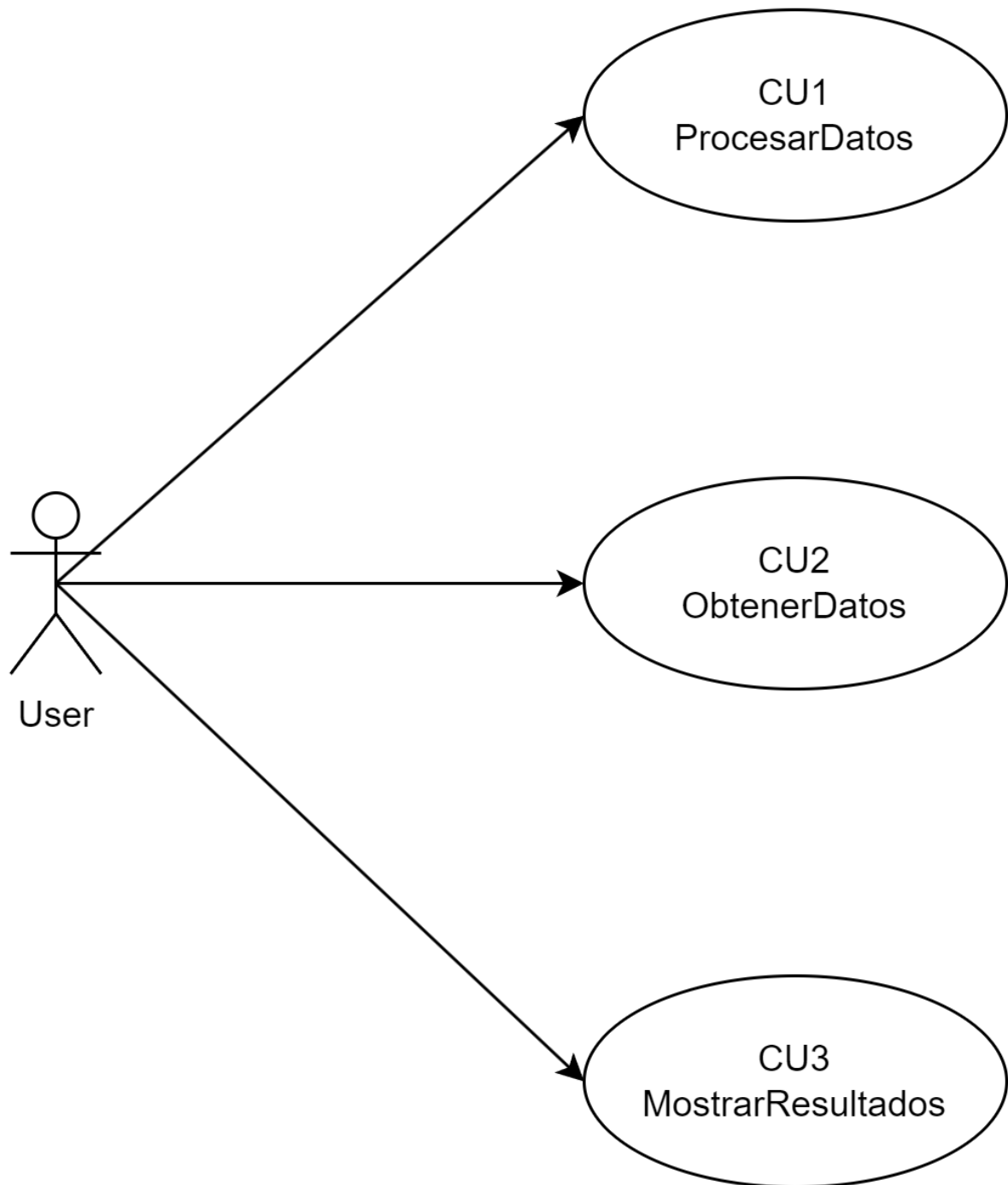




# Escenarios

## Diagrama de Casos de Uso

El siguiente diagrama muestra los diferentes casos de usos del usuario.



# Descripción

## **Caso de uso 1: Procesar Datos**

Para este caso de uso el usuario solicitará el procesamiento de los datos de vídeos de YouTube. Los archivos a procesar deben estar disponibles en la máquina local del usuario. La información se enviará al sistema, leyendo archivo por archivo, que separa los vídeos analizados sobre cada país.

Cuando se envían todos los datos, el sistema responderá con un ID de procesamiento para su posterior consulta y obtención de los datos.

## **Caso de uso 2: Obtener Resultados**

Una vez obtenido el ID de procesamiento, el cliente consultará periódicamente por el resultado del procesamiento. El tiempo entre consultas y el tiempo de espera máximo para obtener los resultados (timeout) será configurable en cada cliente.

El sistema responderá con dos respuestas posibles.

1. Aviso de resultado pendiente
2. Resultados

En caso de recibir un aviso de resultado pendiente, el cliente continuará consultando.

En caso de que haya transcurrido el tiempo máximo de espera, el cliente mostrará un aviso de que no se pudo recuperar la información y finalizará.

Recibir los resultados implica recibir un reporte con la información solicitada por el usuario.

1. Tuplas con información de los videos
  - a. ID
  - b. Título
  - c. Categoría
2. Día de más popularidad
  - a. Formato de Fecha YYYY-mm-ddThh:MM:ssZ
3. Obtener los archivos Thumbnail descargados por el sistema

Una vez recibidos los resultados, se procederá a mostrarlos al usuario.

## **Caso de uso 3: Mostrar Resultados**

Los resultados obtenidos se mostrarán por pantalla en la terminal del usuario y los thumbnail correspondientes estarán descargados en la máquina local del usuario. Se le informará al usuario por pantalla la carpeta en la cual se encuentran los archivos descargados.

# Simulación

Con el objetivo de ver el funcionamiento del Sistema se simulará una consulta a través de las herramientas **docker** y **docker compose** simulando el la corrida del sistema en una máquina host, donde cada servicio simulará las Aplicaciones del Sistema, Cliente y Middleware.

Se proveerá un **script** de lanzamiento el cual será el encargado de

1. Levantar los entornos de docker, tanto el Sistema completo como el Middleware.
2. Levantar el entorno de docker de un cliente y realizar una consulta al Sistema.
3. Imprimir un mensaje de espera, hasta recibir los resultados.
4. Mostrar los resultados por pantalla y la ubicación de los archivos thumbnail descargados.
5. Apagar de manera gracefully todos los entornos utilizados.